

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

First Named

Inventor : Chang-Ning Huang

Appln. No.: 10/662,602

Filed : September 15, 2003

For : CHINESE WORD SEGMENTATION

Docket No.: M61.12-0514

Group Art Unit: 2626

Examiner: Leonard Saint Cyr

**DECLARATION OF INVENTION AND REDUCTION TO PRACTICE
PRIOR TO THE EFFECTIVE DATE OF CITED REFERENCE**

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

1. I, Jianfeng Gao, am listed as an inventor for the above-referenced patent application which was filed September 15, 2003.

2. My understanding is that claims of the above-referenced patent application are being rejected based upon an article attributed to Andi Wu entitled Customizable Segmentation of Morphologically Derived Words in Chinese, which has a publication date of February of 2003, which is prior to the September 15, 2003 filing date of the above-identified application.

3. Attached as Exhibit A (17 pages) is draft report that I prepared (with input from Ashley Chang, Changning Huang, Jianfeng Li, Mu Li, Wenfeng Yang, Ye Zhang, and Xiaoda Zhu) prior to February of 2003. The draft report of Exhibit A describes work that myself and others were conducting at the time when the report was drafted.

I declare that all statements made herein are of my own knowledge and are true; and that all statements that are made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 U.S.C. § 1001 and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Date

2/3/2009

Jianfeng Gao

*** DRAFT ***

A Prototype of Chinese Lexical Services Platform

Jianfeng Gao

(Input from: Ashley Chang, Changning Huang, Jianfeng Li,
Mu Li, Wenfeng Yang, Ye Zhang, and Xiaoda Zhu)

Natural Language Computing Group

Microsoft Research

Beijing, 100080, China

jfgao@microsoft.com

Abstract

This paper reports an ongoing project of Chinese lexical services platform (LSP) at Microsoft research Asia (MSRA). Chinese LSP provides several fundamental features of word-level Chinese language processing including: word segmentation, morphological analysis, factoid detection, and named entity recognition (NER). There are two well-known characteristics of Chinese language: (1) no well-defined lexicon, and (2) no word boundary in written text. In Chinese LSP, we define a Chinese word as (1) an entry in a lexicon, (2) a morphologically derived word, (3) a factoid, or (4) a named entity (NE). We then build a statistical language model to estimate the probability of Chinese word strings. Therefore, given a Chinese character string, Chinese LSP can not only select the word segmentation among all possible segmentations by computing the word string probability based on the language model, but also detect the word category (i.e. certain type of factoid or NE). We will describe the techniques we used, and present experimental results.

1. Introduction

This paper reports an ongoing project of Chinese lexical services platform (LSP) at Microsoft Research Asia (MSRA). This is a sub-project of LSP which is led by natural language group (NLG) at Redmond.

Chinese LSP provides four fundamental features of word-level Chinese language processing (NLP) including: (1) word segmentation, (2) morphological analysis, (3) factoid detection, and (4) named entity recognition (NER). (Other two features i.e. spelling error correction and new word detection, will be addressed in the future version.)

There are two well-known characteristics of Chinese language, which make the Chinese word segmentation much more difficult than English counterpart: (1) no standard definition of word and lexicon, and (2) no word boundary in written text. Therefore, unlike English, we cannot separate the solution to Chinese word segmentation from solutions to the other three features. Instead, we would like to arrive at a unified approach to all the four features.

In Chinese LSP, we define a Chinese word as

- (1) An entry in a lexicon: the lexicon we used contains 98,668 words, including 22,996 Chinese characters stored as single-character words.
- (2) A morphologically derived word, which includes words derived by affixation (e.g. 朋友们 *peng2-you2-men0* 'friend+s'), reduplication (e.g. 高兴 *gao1-xing4* 'happy' → 高高兴兴 *gaol-gaol-xing4-xing4* 'happily'), etc. Detailed description can be found in Section 2.X.
- (3) A factoid such as date, time, etc. Detailed description can be found in Section 2.X.
- (4) A named entity (NE), including personal name (i.e. Chinese personal names such as 毛泽东 *ma2-ze2-dong1* 'Zedong Mao' and transliterated foreign names such as 克林顿 *ke4-lin2-dun4* 'Clinton'), location name such as 上海市 *shang4-hai3-shi4* 'Shanghai', and organization name such as 微软公司 *wei1-ruan3-gong1-si1* 'Microsoft Corporation'. Detailed description can be found in Section 2.X.

We have the above four categories of words because they respectively achieve different functionalities in Chinese NLP, and are detected using different techniques in Chinese LSP. For example, given a Chinese sentence in Figure 1-a, one plausible word segmentation is shown in Figure 1-b, and the output of Chinese LSP is shown in Figure 1-c.

-
- (a) 朋友们十二点三十分高高兴兴到李俊生家吃饭
 (b) 朋友们/十二点三十分/高高兴兴/到/李俊生/家/吃饭
 (c) [朋友+们MA_S] [十二点三十分12:30 FT_TIME] [高兴MR_AABB] [地] [到] [李俊生NE_PN] [家] [吃饭]
-

Figure 1, A Chinese sentence in (a)

We can see that Chinese LSP system not only detects the word boundary, but also detects the word category of each word and process it according as follows:

- (1) If the word is a morphologically derived word, then detect (a) the lexical form of the derived word, e.g. 朋友 and 高兴 are the lexical forms of the words 朋友们 and 高高兴兴, respectively; and (b) the morphological pattern, i.e. the word is derived by affixation such as 朋友们 or reduplication such as 高高兴兴;
- (2) If the word is a factoid, then detect (a) its type TIME or DATE etc. and the normalized form, e.g. 12:30 is the normalized form of 十二点三十分, 十二点半, and 12点30分 etc;
- (3) If the word is a NE, then detect its type e.g. 李俊生 is a personal name;
- (4) Otherwise the word is an entry stored in the lexicon.

Our unified approach is based on statistical language modeling (SLM). We generate an n -gram language model (LM) to estimate the probability of Chinese word strings. Then, given a Chinese character string, by computing the word string probability based on the LM, Chinese LSP can not only select the word segmentation among all possible segmentations but also detect the word category.

To evaluate the performance of Chinese LSP, a Chinese word segmentation spec has been developed at MSRA, based on which a large open test set has been constructed manually. We also address the issue how to compare word segmentation performances across different systems. This is a non-trivial problem because word segmentation spec and lexicon vary from different systems.

In the remainder of this paper, we first describe the techniques we used for word segmentation in Chinese LSP including architecture, lexicalization for morphological analysis, FSA for factoid detect, and class-based LM for NER. In Section 3, we discuss in detail the development of the spec of word segmentation, construction of test set, the evaluation methodology, and present experimental results of Chinese LSP. Finally, we present our conclusions and future work in Section 4.

2. Unified Approach to Chinese Word Segmentation

2.1 Source-Channel Model

In Chinese LSP, we used word/class n -gram models. N -gram LM is a stochastic model which estimates the probability of the word given its previous words. In practice, trigram approximation is widely used, which assumes that the probability of a word w_i is only dependent upon two immediate preceding words w_{i-2} and w_{i-1} , $P(w_i|w_{i-2}, w_{i-1})$.

LM-based Chinese word segmentation can be described as follows: given a Chinese sentence S , which is a character string, for all possible word segmentations W , find the most likely one W^* which achieves the highest probability. That is

$$W^* = \arg \max_W P(W | S) \quad (1)$$

According to Bayes' decision rule, previous research equivalently to Equation (1) performs the following:

$$W^* = \arg \max_W P(W)P(S | W) \quad (2)$$

This is referred to as source-channel approach to speech recognition. In Chinese LSP, following the Chinese word definition described in Section, we define word class C as follows (Detailed definition can be find in Table 1):

- (1) Each type of NE is defined as a class, e.g. all personal names belong to a class denoted by NE_PN ;
- (2) Similarly, each type of factoid is defined as a class, e.g. all time expressions belong to a class denoted by FT_TIME ;
- (3) For words stored in the lexicon, each word is defined as a class;

Notice that we do not handle morphologically derived words in our model. We will come back to this problem in Section 2.X. Based on the above definitions of classes, we convert the word-based models in Equation (2) into class-based models as shown in Equation (3)

$$C^* = \arg \max_C P(C)P(S | C). \quad (3)$$

Here, $P(C)$ is the language model estimating the probability of word class sequence. For a certain word class such as personal name NE_PN , $P(C)$ indicates how likely it occurs in a certain context. For example, NE_PN is more likely to occur after a title such as Prof. and Mr. So $P(C)$ is also referred to as *context model* afterwards. $P(S|C)$ is a generative model estimating how likely a character string is generated given a certain word class. For example, the character string 李俊生 *li3-jun4-sheng1* 'Li Junsheng' is more likely to be a personal name than 里俊生 *li3-jun4-sheng1* 'Li Junsheng' because 李 is a very common family name in China. So $P(S|C)$ is also referred to as *class model* afterwards. Notice that we used one trigram model for context model whereas different classes have different class models as shown in Table 2. The detailed description will be presented in the remainder of this section.

2.2 Search Strategy

Given the decision rule of Equation (4), the overall architecture of Chinese LSP is summarized in Figure 2. Pre-processing does sentence breaking whereas post-processing does morphological analysis. Given a Chinese sentence, which is a string of Chinese characters, first, all possible words together with class tag C and class model probability $P(S|C)$ are generated in the lattice; second, Viterbi search (or A^* search) is used to find the best (or best- N , where $N > 1$) word segmentation(s) with the highest probability $P(C)P(S|C)$.

Considering derived words such as NE and factoid, a character string of any lengths can be a word candidate of various class tags, so a set of pruning strategies are necessary for generating word candidates in the lattice to keep the search space manageable. The amount of search is controlled by log-probability score given by class models $P(S|C)$.

For different classes of words, we used different class models to estimate class probability $P(S|C)$. Given a certain S , all candidates (hypotheses) are ranked by $\log(P(S|C))$ the amount of candidates C is controlled by two parameters:

- number threshold – the maximum number of hypotheses cannot be larger than a given threshold;
- log-probability threshold – the difference between the log-probability score of the top-ranked hypothesis and the bottom-ranked hypothesis cannot be larger than a given threshold.

Given a string of characters S^* , we generate word candidates with different classes as follows:

- Lexical words: for any substring $S \subseteq S^*$, we assume $P(S|C) = 1$ and tagged the class as lexical word if S forms an entry in the lexicon, Otherwise $P(S|C) = 0$; that is, we do not consider OOV.
- Factoid words: For each type of factoid, we generate a grammar G , which is a FSA. We assume $P(S|C) = 1$ if S can be parsed successfully using G , $P(S|C) = 0$ Otherwise. Given S^* , we used the maximum matching method to generate factoid words in the lattice. The method involves starting at the beginning of S^* , finding the longest factoid word starting at that point, and then repeating the process starting at the next character until the end of the S^* is reached.
- Named entity words: we consider only four types of named entity words: personal names (PN), location names (LN), organization names (ON) and transliterations of foreign words (FN). We used different class models for different type of named entities as described in Section XX.
- Morphologically derived words are identified in the post-processing phase. Given a segmented Chinese sentence C , which is a string of words (including lexical words, factoid words, and/or named entities), we used the maximum matching method to find the morphologically derived words in the sentence by looking up a morph lexicon containing common-used morphologically derived words. In Section XX, we will describe in detail the method of constructing the morph lexicon.

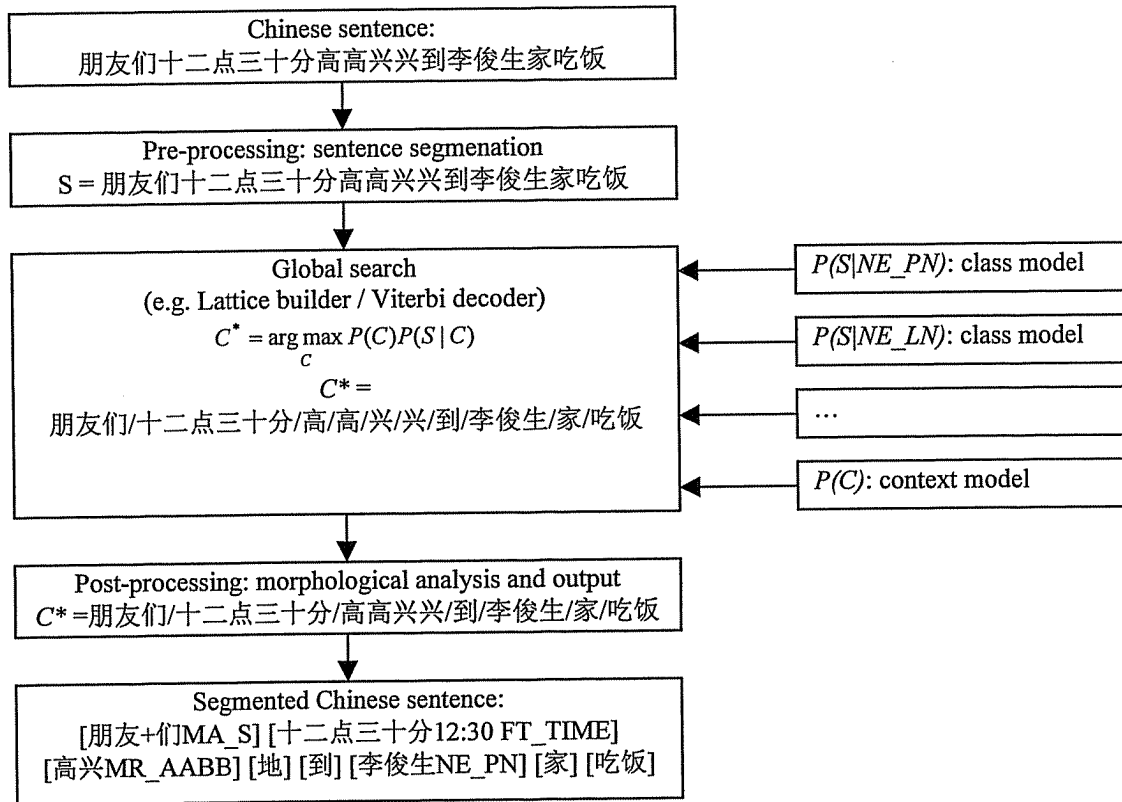


Figure 1: Architecture of Chinese LSP prototype

3. Named Entity

For each NE, we will describe its class model and how to generate NE candidate in the lattice.

3.1 Chinese personal names

A typical Chinese personal name PN consists of a family name F and a given name G . It is always of the form $F+G$. The family name set is restricted: there are a few hundred single-character family names and about ten double-character ones. Given names are most commonly one or two character long. The difficulty of Chinese person name recognition is that given names can consist, in principle, of any characters or pair of characters, though some characters are certainly more likely than others.

Assume that a Chinese person name is of the form $PN=FG_1G_2$ or $PN=FG$. We used a generative model to estimate the probability of a given character string S to be a person name, $P(S|PN)$. We assume that S consists of a family name sub-string S_F and a given name sub-string S_G , where both S_F and S_G are of the length one-two characters. In particular, for $PN=FG_1G_2$, we denote S_{G1} and S_{G2} as the first and second given name character, respectively. We decompose the generation of PN into three (for $PN=FG_1G_2$) or two (for $PN=FG$) steps:

- (1) Generate the family name sub-string S_F , with the probability $P(S_F|F)$;
- (2) Generate the (second) given name sub-string S_G , with the probability $P(S_G|G)$ (or $P(S_{G2}|G_2)$);
- (3) Generate the first given name given name, with the probability $P(S_{G1}|S_{G2}, G_1)$.

For example, the probability of the string 李俊生 *li3-jun4-sheng1* ‘Li Junsheng’ to be a Chinese person name PN would be estimated as $P(\text{李俊生}|PN) = P(\text{李}|F)P(\text{生}|G_2)P(\text{俊}|生, G_1)$.

The probabilities can be estimated using the Chinese personal name list by maximum likelihood estimation (MLE). Now, let us describe the methods of dealing with the data sparseness problem – how to estimate the probabilities for unseen family name and given name characters or character-pairs.

From the Chinese personal name list we used for training, we collected a Chinese family name list SF contains 373 entries. We then simply assign $P(S_F|F)=0$ if S_F is not included in SF .

If a character-pair is unseen as a double-character given name, we backoff bigram probability $P(S_{G1}|S_{G2}, G_1)$ to the unigram probability $P(S_{G1}|G_1)$ using the modified absolute discounting method [XX].

If a character is unseen as a given name character, we assigned $P(S_G|G)=0$ when SG is a commonly-used single-character words (we used a list of 200 most commonly used Chinese single-character words); whereas we used Good-Turing estimate for other unseen given name characters. Good-Turing estimate assumes that the number of unseen events is same as the number of the events occur once. The final estimate for unseen given name characters is $P(S_G|G)=1/(N+NI)$, where N is the total number of given name character occurrences in the training data, and NI is the number of given name characters which occur once in the training data.

3.2 Location names

Unlike Chinese personal names, location names LN have no well-defined pattern although some LN end with a so-called LN keyword such as 市 *shi4* ‘city’ and 省 *sheng3* ‘province’. The LN keyword list we used contains 100 entries. Therefore, LN can be, in principle, of any length and of any characters. To seek the best tradeoff between precision and recall, the LN class model we used is a hybrid model which combines a set of heuristic rules H and a character-based bigram model. Given a sentence S^* , H is first used to restrict the generation of LN candidates, and then the bigram model is used to estimate the probability of any LN candidate, which is a substring S of S^* , to be a LN , $P(S|LN)$. According to H , given a , the substring S can be a LN candidate if and only if S meets at least one of the following three descriptions:

- S is an entry stored in certain LN lexicon. The LN lexicon we used contains 30,000 Chinese and foreign location names such as 北京 *bei3-jing1* ‘Beijing’, and 纽约 *nui3-yue1* ‘New York’.
- S ends with a LN keyword, and the remaining characters of S , refer to as a substring S' , can be segmented into words only stored in the LN lexicon. That is, for all possible segmentations of S' , there is at least one which contains words stored in the LN lexicon only. For example, 北京市

bei3-jing1-shi4 ‘Beijing city’ is a LN candidate because the string ends with the LN keyword *shi4* ‘city’, and the remaining string contains only one word 北京 *bei3-jing1* ‘Beijing’ which is stored in the LN lexicon. However, the string 北京和上海市 *bei3-jing1-he2-shang4-hai3-shi4* ‘Beijing and Shanghai city’ is not a LN candidate because given the segmentation 北京/和/上海/市, and *he2* ‘and’ is not a word in the LN lexicon.

- *S* ends with a LN keyword, and the remaining substring *S'*, considering the segmentation of *S**, contains only single-character words which are not LN keywords. For example, given the sentence with its segmentation /我们/到达/夏/米/尔/河/地区/ *wo3-men2-dao4-da2-xia4-mi3-er3-he2-di2-qul* ‘We arrived at the area of Shamir river’, because 河 *he2* ‘river’ is a LN keyword, the LN candidates include 夏米尔河, 米尔河, and 尔河.

If *S* is a LN candidate, then the probability $P(S|LN)$ is estimated using a character bigram model. For example, given the string *S*=夏米尔河 *xia4-mi3-er3-he2* ‘Shamir river’, which is a LN candidate according to *H*, the probability of *S* to be a LN would be estimated as $P(\text{夏米尔河}|LN) = P(\text{夏}|\langle LN \rangle)P(\text{米}|\langle LN \rangle, \text{夏})P(\text{尔}|\text{夏}, \text{米})P(\text{河}|\text{米}, \text{尔})P(\langle LN \rangle|\text{尔}, \text{河})$, where $\langle LN \rangle$ and $\langle /LN \rangle$ are symbols denoting the beginning and the end of a LN, respectively.

The bigram probabilities are trained on a LN list using MLE. Bigram would backoff to unigram if the character pair does not occur in the LN list. Good-Turing method is used to estimate the unigram of unseen character.

Our experiments show that the hybrid model achieved the best tradeoff between precision and recall. However, there are some problems of using heuristic rules to filter unlikely candidates. The most serious one is that all LN without LN keyword would loss if they are not stored in the LN lexicon. For example, the string 夏米尔 *xia4-mi3-er3* ‘Shamir’ which would not be a LN candidate according to *H*, can be a name of a city in a sentence without the LN keyword 市 *shi4* ‘city’ following it. The foreign name class model and the cache model described in Sections XX and XX will remedy the problem to some degree.

3.3 Organization names

Organization names ON are the most difficult kind of NE to be recognized because (1) similar to LN, ON has no well-defined patterns; and (2) ON is usually a nested NE which contains other NE such as PN and LN, e.g. the ON 中国国际航空公司 *zhong1-guo2-guo2-ji4-hang2-kong1-gong1-si* ‘Air China Corporation’ contains the LN 中国 *zhong1-guo2* ‘China’.

For the first problem, we used the similar approach we used for LN. We generate an ON keyword list which contains 1,355 entries such as 大学 *da4-xue2* ‘university’ and 公司 *gong1-si* ‘corporation’, and assume that a character string can be an ON candidate only if it ends with an ON keyword. Although the use of ON keyword list achieves higher precision, it

For the second problem, we first segment the ON candidate *S*, which is a character string ending with an ON keyword, into word/class string *C* using the word segmentation system as shown in Figure XX except that (1) the models (class models and context model) are trained based on the segmented ON list, and (2) we do not use ON class model. Then, the probability of *S* to be ON $P(S|ON)$ is estimated using a class-based bigram model. Taken the segmented word/class strings *C* as hidden variables, $P(S|ON)$ can be formally estimated as

$$P(S|ON) = \sum_C P(S, C|ON) = \sum_C P(C|ON)P(S|C, ON),$$

where *C* can be any possible segmentations of *S* given the models trained on the ON list. Since $P(S|C, ON) = P(S|C) = 1$, we have

$$P(S|ON) = \sum_C P(C|ON).$$

Using the so-called maximum approximation, it can be rewritten as

$$P(S | ON) \approx \max_C P(C | ON) \quad (3)$$

For example, given the ‘most likely’ segmented string $C=\text{中国/国际/航空/公司}$ of $S=\text{中国国际航空公司}$, the probability of S to be a LN would be estimated as $P(\text{中国国际航空公司} | ON) \approx P(\text{中国/国际/航空/公司} | ON) = P(\text{中国} | <ON>)P(\text{国际} | \text{中国})P(\text{航空} | \text{国际})P(\text{公司} | \text{航空})P(</ON> | \text{公司})$, where $<ON>$ and $</ON>$ are symbols denoting the beginning and the end of a ON, respectively.

The bigram probabilities are trained on a segmented ON list using MLE. Bigram would backoff to unigram if the word/class pair does not occur in the LN list. Good-Turing method is used to estimate the unigram of unseen word/class.

3.4 Transliterations of foreign names

Foreign names FN are usually transliterated using Chinese character string whose sequential pronunciation mimics the source language pronunciation of the name. Since foreign names can be of any length and their original pronunciation is effectively unlimited, the recognition of such names is tricky. Fortunately, there are only a few hundred Chinese characters that are particularly common in transliterations, such as 克林顿 *ke4-lin2-dun4* ‘Clinton’. We used a transliterated name list TNL containing 618 Chinese characters. In addition, two symbols $<FN>$ and $</FN>$ which denote respectively the beginning and the end of a FN are added to TNL. We assumed that a character string S can be a possible foreign name only if all characters of the string belong to TNL. The probability of S to be a FN, $P(S | FN)$ is estimated using a character-based trigram model.

For example, the probability of the string 克林顿 *ke4-lin2-dun4* ‘Clinton’ to be a FN would be estimated as $P(\text{克林顿} | FN) = P(\text{克} | <FN>)P(\text{林} | <FN>, \text{克})P(\text{顿} | \text{克林})P(</FN> | \text{林顿})$.

The trigram probabilities are trained on a FN list using MLE. We used the recursive backoff scheme to deal with the data sparseness problem: trigram backoff to bigram, bigram backoff to unigram. As for the unigram estimate of an unseen character, if it is not included in the TNL, we assigned zero probability, otherwise we used Good-Turing estimate.

Notice that a FN can be a PN, a LN, or an ON. For example, we know that 肯尼迪 *ken3-ni2-di2* ‘Kennedy’ is a LN in the context 飞机抵达肯尼迪机场 *fei1-ji1-di3-da2-ken3-ni2-di2-ji1-chang3* ‘The airplane arrived at Kennedy airport’, and is a PN in the context 肯尼迪发表演说 *ken3-ni2-di2-fa1-biao3-yan3-shuo1* ‘Kennedy gave a speech’. Since it is very difficult to differentiate among the three categories without context information which is captured by context model, whenever we detect a possible FN given a string S , i.e. $P(S | FN) > 0$, we generate three named entity candidates, each for one category, in the lattice and assign them the same class model probability, i.e. $P(S | PN) = P(S | LN) = P(S | ON) = P(S | FN)$. In another word, we delay the determination of its category until decoding where the context model is used. By doing so, it also remedies, to some degree, the problem of losing LN and ON candidates due to the use of H as mentioned in previous sections.

3.5 Cache model

In a document, same NEs usually occur more than once. Therefore, if a certain NE was identified, the probability of its occurrence in the remainder of the same document would be increased. To take advantage of this observation, we used multiple cache models, each model for one type of NE. Whenever an NE, which is a character string S , is identified, we push S into the corresponding NE cache. The cache probability of a S to be an NE, $P_{\text{cache}}(S | NE)$ is calculated using unigram estimate:

$$P_{\text{cache}}(S | NE) = \frac{C(S)}{N},$$

where $C(S)$ is the number of occurrence of S in the NE cache, and N is the size of the cache. When generating the lattice, we assign for each NE candidate $P(S | NE)$ by linear interpolating the cache

probability $P_{cache}(S|NE)$ and the NE class probability $P_{static}(S|NE)$ assigned using the static NE class models described in the previous sections:

$$P(S|NE) = \lambda P_{static}(S|NE) + (1 - \lambda) P_{cache}(S|NE),$$

where $\lambda \in [0,1]$ is the interpolation weight, optimized on the held-out data set.

3.6 Abbreviation

We found that many errors result from the occurrence of abbreviation. For different kinds of NEs, different strategies are adopted to deal with abbreviations. For PN recognition, if Chinese surname is followed by the title, then this surname is tagged as PN. For example, 左校长 *zuo3-xiao4-zhang3* ‘President Zuo’ is tagged as <PN>左</PN> 校长. For LN recognition, if at least two location abbreviations occur consecutive, the individual location abbreviation is tagged as LN. For example, 中日关系 *zhong1-ri4-guan1-xi4* ‘Sino-Japan relation’ is tagged as <LN>中</LN><LN>日</LN> 关系. For ON recognition, if an organization abbreviation is followed by LN, which is again followed by organization keyword, the three units are tagged as one ON. For example, 中共北京市委 *zhong1-gong4-bei3-jing1-shi4-wei3* ‘Chinese Communist Party Committee of Beijing’ is tagged as <ON>中共<LN>北京</LN>市委</ON>. At present, we collected 112 organization abbreviations and 18 location abbreviations.

4. Morphological Analysis

The morphologically derived words we handled in Chinese LSP include the following five types:

1. **Affixation** such as 朋友们 *peng2-you3-men0* (friend + plural) ‘friends’ which is derived by adding the plural suffix 们 *men0* to the noun 朋友 *peng2-you3*.
2. **Reduplication** such as AABB reduplication like 高兴 *gao1-xing4* ‘happy’ → 高高兴兴 *gao1-gao1-xing4-xing4* ‘happily’, and A-not-A question formation like 高兴 *gao1-xing4* ‘happy’ → 高不高兴 *gao1-bu4-xing4-xing4* ‘happy?’.
3. **Merging** such as 上班 *shang4-ban1* ‘on duty’ + 下班 *xia4-ban1* ‘off duty’ → 上下班 *shang4-xia4-ban1* ‘on-off duty’.
4. **Head particle** such as the expressions that are verb + comp, e.g. 走 *zou3* ‘walk’ + 出去 *chu1-qu4* ‘out’ → 走出去 *zou3-chu1-qu4* ‘walk out’.
5. **Split** is a set of expressions that are separate words at the syntactic level but single words at the semantic level, such as bi-character words split by particles such as 了 (吃饭+了 → 吃了饭) 和 不 (看见+不 → 看不见).

The morphological analysis of western languages such as English can be handled using well-known techniques from finite-state morphology (CITE XXX), but they are very difficult to be extended to handle Chinese language due to the following two reasons. First, Chinese morphological rules are not as general as the English counterpart. For example, almost all English plural nouns can be generated using the rule ‘noun + s → plural noun’. However in Chinese, only a portion of plural nouns can be formed using its Chinese counterpart ‘noun + 们 → plural noun’ such as 朋友们 *peng2-you3-men0* ‘friends’ whereas others cannot such as 南瓜们 *nan2-gua1-men0* ‘pumpkins’. Second, the operations required by Chinese morphological analysis, such as copying in reduplication, merging and splitting, cannot be implemented using the current finite-state networks. For the first problem, Sproat et al. (1996) used a stochastic finite-state transducer. For example, they represent the fact that 们 attaches to nouns by allowing empty transitions from the final states of all noun entries in the lexicon, and then estimate the cost of the derived words using text corpus i.e. the cost of 南瓜们 would be much higher than that of 朋友们 because the former rarely occurs in the text. However their method can only handle the affixation case.

Our method is the so-called lexicalization. That is, we simply expand out morphologically derived word forms of the five abovementioned types and incorporate them into the lexicon. The procedure involves

three steps: (1) morphologically derived word-item generation, (2) linguistic selection, and (3) statistical selection.

4.1 Morphologically derived word-item generation

The morphologically derived word list is generated from two sources: (1) a dictionary, and (2) a corpus of a million sentences. The dictionary we used contains morphological attributes for each word. For example, it indicates that the noun 朋友 can be expanded to be a plural noun form by adding an affix morpheme 们, but the noun 南瓜 cannot. We expand out all morphologically derived word indicated in the dictionary and add them into the lexicon directly. We then extract more derived words from the corpus using a set of morphological rules which are compiled by Chinese linguist. From the derived words, we get rid of the entries occur less than three times in the corpus. The resulting list containing about 50,000 items is ready for linguistic and statistical selection described below.

4.2 Linguistic selection

Linguistic selection is produced manually. We have three Chinese native speakers manually check each item of the list. Since the purpose of generating the morphologically derived words is to add words to the lexicon for word segmentation, the basic guideline for the checking is to make sure what are included in the **Correct** category are morphologically generated **new words** rather than expressions that are modifier + modified or syntactically motivated, productive combination of already existing words. The detailed description of the criteria used in the checking can be found in Appendix XX.

In particular, assessors would check for each item to see if the following linguistic attributes are correct: (1) the generated surface form, (2) lexicon form, (3) pattern, and (4) POS. After checking, the original word list grouped into three categories:

- **Correct:** these entries should be included in the lexicon.
- **Error:** these are wrongly extracted entries, and should not be included in the lexicon.
- **Tentative:** these are open for discussion from linguistic point of view. Detailed description can be found in Appendix XX.

The words in the **Tentative** category are debatable items. This is due to the fact that there is no standard definition of word in Chinese and the definition of word usually varies from different applications. For example, a large portion of these words are expressions that are separate words at the syntactic level but single words at the semantic level. For instance, 签了字 *qian1-le0-zi4* ‘already signed’ are normally segmented into three words in word segmentation, but in logical form they function as one word meaning 签字 *qian1-zi4* ‘signed’ + 了 *le0* ‘already’. Similarly, 跳起舞来 *tiao4-qi3-wu3-lai2* ‘begin dancing’ appears as 跳/起/舞/来 in segmentation but can be restructured to become 跳舞 *tiao4-wu3* ‘dancing’ + 起来 *qi3-lai2* ‘begin’. For LSP, 签了字 can be segmented into three words, but in the future there should be some kind of mapping between 签了字 and 签字 + 了, so that we can relate the surface form to the underlying form.

This may be useful to information retrieval or question/answering, for example, where we should be able to retrieve 签字 when the query contains 签了字. This may also be important to machine translation, where it might be a mistake to translate 签 and 字 separately.

4.3 Statistical selection

Since the words in the **Tentative** category cannot be judged using linguistic features, we select them using a set of statistical features obtained on a large corpus. We therefore convert the problem of Chinese word selection into the problem of estimating the conditional probability that an morphological derived item m is a word w , given its statistical feature vector $F=(f_1, \dots, f_p)$. This is denoted as $P(m \rightarrow w|F)$. In what follows, we will describe (1) the selection of feature set, and (2) the estimation of the probabilistic model $P(m \rightarrow w|F)$.

We used an approximate information gain-like metric (Gao et al., 2002) consisting of three statistical features, namely (1) *mutual information*, (2) *context dependency*, and (3) *relative frequency*. The basic idea

behind the metric is that a Chinese word should appear as a stable sequence in the corpus. That is, the components within the word are strongly correlated, while the components at both ends should have low correlations with outer words. This is illustrated in Figure XX.

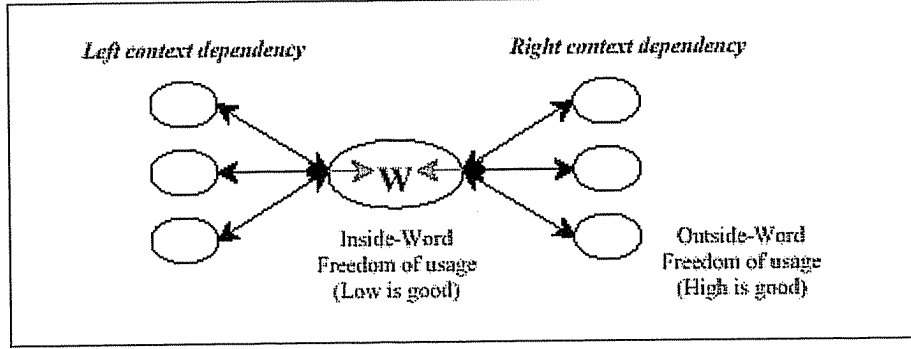


Figure XX. The mutual information and context dependency of a word

Mutual information (MI) is a criterion to evaluate the correlation of different components (e.g. characters or short words) in the word. For example, let $MI(x,y)$ denotes the mutual information of a component pair (x, y) . The higher the value of MI , the more likely that x and y are to form a word.

Context dependency (CD) is a criterion to evaluate the correlation of the candidate word and components outside at both ends. The value of left CD can be estimated by

$$LSize = |L|$$

or

$$MaxL = MAX_{\alpha} \frac{f(\alpha X)}{f(X)}$$

where $f(.)$ is frequency, L is the set of left adjacent strings of X , $\alpha \in L$ and $|L|$ means the number of unique left adjacent strings. Clearly, the higher the value $LSize$, or the lower the value of $MaxL$, the more likely the item has left CD. Similarly, the value of right CD can be estimated by

$$RSize = |R|$$

or

$$MaxR = MAX_{\beta} \frac{f(X\beta)}{f(X)}$$

where $f(.)$ is frequency, R is the set of right adjacent strings of X , $\beta \in R$ and $|R|$ means the number of unique right adjacent strings. The extracted word should have neither left nor right *context dependency*.

Relative frequency (RF) is a criterion to reduce noise in the lexicon. All words with lower frequency are removed from the lexicon.

Given the feature set, the probability $P(m \rightarrow w|F)$ can be estimated using probabilistic models such as maximum entropy or logistic regression. The remaining problem is how to obtain positive and negative examples for training. We first extract all character bigrams from a text corpus. We then treat all bigrams that stored in the lexicon as bi-character word as positive examples, and treat the others as negative examples. We therefore use a simple thresholding algorithm for word selection. It involves three steps:

1. Select a threshold θ .
2. For each item m , compute the probability $P(m \rightarrow w|F)$.
3. Select each item whose probability to be a word is large than θ , and add it into the lexicon.

The threshold is selected experimentally to achieve the best tradeoff between precision and recall.

5. Factoid

The types of factoid handled in Chinese LSP are shown in table XX. For each type of factoid, we generate a grammar G , which is a FSA. We assume $P(S|C) = 1$ if S can be parsed successfully using G , $P(S|C) = 0$ Otherwise. Given S^* , we used the maximum matching method to generate factoid words in the lattice. The method involves starting at the beginning of S^* , finding the longest factoid word starting at that point, and then repeating the process starting at the next character until the end of the S^* is reached.

We used FSA because (1) the detection of factoid is context free¹; (2) FSA is deterministic so it is very efficient; and (3) FSA is decidable so it is easy to maintain. The types of factoid in table XX are defined in a way that the granularity of factoid is fine enough not only for various applications LSP would support but also for FSA to handle. To justify the effectiveness of our FSA-based approach to factoid detection, we evaluate the precision and recall of the four types of factoid (i.e. Money, Percentage, Date, and Time) using the MET-2 test set (CITE XXX). The results are shown in Table XX. For comparison, we also list the results reported by National Taiwan University (NTU) presented at (CITE XXX)².

FSA grammar is compiled by Chinese linguists. To improve the effectiveness of FSA generation and maintenance, we developed a set of authoring toolkit. It includes the following three components: (Detailed description can be found in Appendix XX)

- **FSA editor** is an editor for linguist to edit factoid rules which are written using regular expression, and provide grammar check.
- **FSA compiler** converts factoid rules from regular expression into FSA of binary format, and optimize the resulting FSA through the process of determination and minimization.
- **FSA runtime** provides a set of APIs for Chinese LSP to load FSA files and detect factoid candidates given a character string on the fly.

#	Type	Example
DATE	Date	
TIME	Time	
PERC	Percentage	
MNEY	Money	
NUM	Number	
MEA	Measure	
EMIL	Email	
PHNE	Phone number	
URL	WWW	

Figure XX. The types of factoid in Chinese LSP

		Precision	Recall	F-Score
Money	NTU*	0.98	0.98	0.98

¹ This is not exactly true, but we have this assumption for simplicity.

² We choose NTU results for comparison because NTU was the only group that reported results of the four types factoid detection at the conference (NAME of the CONF.?).

	FSA	1.00	1.00	1.00
Percentage	NTU	0.98	0.83	0.91
	FSA	0.96	1.00	0.98
Data	NTU	0.88	0.94	0.91
	FSA	0.92	0.93	0.93
Time	NTU	0.70	0.98	0.84
	FSA	0.84	0.88	0.86

Figure XX. Comparison results of factoid detection

6. Model Training

As shown in Figure1, there are two types of models in Chinese LSP: (1) class models and (2) context model. The class models are either trained using NE list such as NE class models or determined using pre-defined grammars such as factoid FSA. We focus our discussion on context model training in this section.

Context model estimates the probability of any class sequences so an annotated text corpus is needed for training. The corpus should be annotated in a way that all words including lexical words, factoid, and NE are segmented and identified³. The manual generation of such corpus is very time-consuming and almost impossible. Given class models, we have two EM-like iterative methods of obtaining the context model:

1. Use a small hand-annotated seed corpus to get an initial context model, then re-annotated the corpus and obtain a more reliable model. Repeat the process until optimal performance is achieved.
2. Use a greedy segmentor i.e. maximal matching (MM) to annotate the corpus, obtain an initial context model on the annotated corpus, then re-annotate the corpus using models, and re-train the context model using the re-annotated corpus. Repeat the process until optimal performance is achieved (Gao et al. 2002).

The hand-annotated seed corpus is under development. We focus on the second method in this section. The greedy segmentor is developed to annotate the corpus. It is an augment of MM segmentor and consists of several serial steps. In each step, it either divides the sentence into smaller pieces or identifies word boundaries using maximum matching principle. These steps are:

1. Divide the sentence using punctuation cures, especially pair punctuations like “ ”, ‘ ’, (), { }, [], etc.
2. Extract factoid words using factoid FSA such as Date, Time, Number, etc. If two or more overlapped factoid words are detected, the one with longer character string is chosen.
3. Segment remaining words by MM using the lexicon. All OOV words such as named entities are segmented into single-character words.
4. A rule-based NER system is used to identify named entities. If two or more overlapped named entities are recognized, the one with longer character string is chosen.

The segmentor is greedy because it deals with the ambiguities in segmentation by maximum matching principle, possibly augmented with further heuristics (e.g. Ming, 2000). For a sequence of characters to be segmented, the method attempts to segment them while keeping the number of resulting words minimal (or using words with maximal lengths.)

³ We do not include morphologically derived words because as mentioned in Section XX, such words are identified in the post-processing phase.

There are two main problems of the greedy segmentor. The first is the so-called the problem of overlapping ambiguity string, referred to as OAS afterwards. Given a string containing three characters ABC, if it can be segmented into either AB+C or A+BC depending on different context, the string is called OAS. Although OAS can be any length, three-character OAS is the most common one. If we used only one MM method e.g. forward MM, whenever there is a string ABC, it is always segmented into AB+C. This thus hurts the performance of the resulting context model. We resolve this problem by using both forward MM and backward MM in the greedy segmentor. Given a character string, we segment it into words using both MM methods and train the context model using both segmented corpus. Experiments showed that comparing to using the forward MM alone; using both MM methods can reduce the number of OAS on the test set from 1220 to 305.

The second problem lies in the NE tagging errors in the annotated corpus due to the poor performance of the rule-based NER system we used. We presented two methods. First, we found in our experiments that the EM-like iterative method described above can increase the NER performance. We test the method for personal name and location name recognition on the test set. Table XX shows the precision and recall of personal name and location name recognition versus number of iterations, where the initial context model at iteration 0 was bootstrapped by segmenting the training text into words using the greedy segmentor i.e. personal names and location names are recognized by the rule-based NER system. It can be observed that the NER performance increases after each iteration, and begins to saturate at the second iteration. The other method is that we can combine multiple NER systems by for example, voting and co-training. Recent research shows that the proper ensemble of multiple weak learners would achieve a strong learner. This research is still on-going.

Iteration	PN		LN	
	Precision	Recall	Precision	Recall
0	70.5	69.4	50.4	67.6
1	89.1	77.2	84.0	79.5
2	89.1	77.2	84.3	79.8

Table XX. NER precision/recall versus for 0-2 iterations

6.1 Comments on Combining Context Model with Class Models

<spoken language processing> For each word type, there is a lamda

7. EVALUATION

In this section, we present the evaluation of the current Chinese LSP system, in four parts. The first is a description of the standard test set and the evaluation criteria we used in our experiments. The second evaluates the overall performance of the system in terms of overall segmentation accuracy, precision-recall of NER, factoid detection, and morphological analysis. The third compares our system with others. The fourth is an evaluation of the system's ability to mimic humans at the task of segmenting text into word-sized units. We call this the naturalness evaluation. This follows Sproat et al.'s work in 1996.

7.1 Evaluation Methodology

A standard test set has been developed at MSRA for evaluating the performance of our Chinese word segmentation system – henceforth LSP. There are several questions we have to answer when we were developing the test set.

1. How large the standard test set would be in order for a reliable evaluation?

2. Does the segmentation in the standard test set depend on a particular base lexicon? If so, how to deal with OOV?
3. Should we assume that there is a single correct segmentation for a sentence?
4. What are the evaluation criteria?
5. Can we make a fair comparison across different systems using the standard test set?

First of all, to achieve a reliable evaluation, the test corpus must be independent, balance, and large enough. Independent means that the test set and the data set we used for training do not overlap. Balance means that the test set should contain documents of the application(s) with wide variety of domain, style, and time. The test set we used contains articles published in 1997's People-Daily. The domain/style distribution is shown in Table 7.1. The test set consists of XX sentences containing approximately 1 million Chinese characters.

Table 7.1 Domain/style distribution for the test corpus.

The standard test set has been developed by annotating the test corpus. The test corpus is first segmented by humans into words with each word attached by its word type including lexicon word, factoid, and named entity. The resulting annotated test set is called base-tagged set, denoted by **BT**. Some fragments are shown in Figure 7.1 (a). Morphologically derived words are then tagged on top of **BT**. The resulting annotated test set is called morph-tagged set, denoted by **MT**. Some fragments are shown in Figure 7.1 (b). In what follows, we only discuss **BT**. The statistics of the standard test set (i.e. both BT and MT versions) are shown in Table 7.2.

(a) Fragments of BT	(b) Fragments of MT
Figure 7.1 Fragments of the standard test set, where the annotation tagset are listed in Appendix X.	

Word Type	Number of tokens	%
Lexicon words (in BT)	205162	83.0
Lexicon words (in MT)		

Morphologically derived words (in MT)			
Named Entity	Personal names	4347	1.8
	Location names	5311	2.1
	Organization names	3850	1.6
Factoid	Date	2310	3.2
	Time	227	
	Percentage	311	
	Money	350	
	Number	3387	
	Measure	451	
	Email	0	
	Phone	13	
	WWW	0	
Punctuations		20250	8.2
Others		265	1.0
Total (in BT)		247039	100
Figure 7.2. The statistics of the standard test set			

The second and the third questions can be answered together. The standard test set is segmented depending on a base lexicon which contains 98,668 words including 22,996 Chinese characters stored as single-character words. For OOV, the current Chinese LSP system can only handle NE and factoid, and leave others as strings of single-character words. We claim that the generation of **BT** is lexicon dependent because of two reasons. First, we allow multiple correct segmentations for a sentence where there are new words. For example, in sentence (2) of Figure 7.1 (a), 森警 *sen1-jing3* ‘wood police’ is a new word not stored in the base lexicon, so in evaluation, either segmenting it as one word or segmenting it into two single-character words is a correct segmentation. The second is compound word. Compound words such as 不法分子 *bu4-fa3-feng4-zi3* ‘badman’ and 心理咨询 *xin1-li3-zil-xun2* ‘’. There is no agreement among Chinese linguists whether or not they should be segmented as one word.

In sentence (X) of Figure XX, XXXX is a compound word that can be segmented as one word or two words XX and XX by humans. If both XX and XX are words stored in the lexicon, either segmentation is correct in evaluation.

We used multiple evaluation criteria. Some are used only for evaluating our system while others can be used to compare word segmentation performance across systems. These criteria are summarized in Table XX. Here, OAS is defined as follows: Given a string containing three characters ABC, if it can be segmented into either AB+C or A+BC (but cannot be both) given a certain context (but cannot be both) given a certain context, the string is called OAS; and CAS is defined as follows: Given a character string ABCD, if it can be segmented into either AB+CD as two words or ABCE as one word (but cannot be both) given a certain context, the string is called CAS.

While all listed criteria in Table XX can be used to evaluate our system, only a portion of them can be used to compare our system with others. In Section XX, the comparison results were reported only on NE precision/recall and # of OAS because these criteria are lexicon independent and there is always a single unambiguous answer of them. CAS is not used because there are a lot of debatable words or compound words such as 不法分子 *bu4-fa3-feng4-zi3* ‘badman’ and 秉公执法 *bin3-gong1-zhi2-fa3* ‘execute the law

justly'. There is no agreement among Chinese linguists whether or not they should be segmented as one word. Factoid precision/recall is not used for comparison because the definition of factoid highly varies from system to system.

<Table XX, evaluation criterion>

7.2 System Results

The training set for context model is collected from XX, containing XX sentences (XX characters). The Chinese LSP system is defined in a way that all components such as factoid detection and NER can be 'switched on/off', so that we can demonstrate the relative contribution of each component to the overall word segmentation performance. The results are shown in Table XX. Notice that the results shown in row 2 are obtained on BT test set using the baseline of our system, where only base lexicon is used. From row 3 to row 6, components are switched on in turn by activating corresponding class models. It can be seen that the overall segmentation accuracy increases as more class models are used. Row 7 shows the results of morphological analysis on MT test set. For comparison, we also include in Table XX the results of using forward maximum match algorithm: proceed through the sentence from left to right, taking the longest match with the base lexicon entry at each point.

<Table XX. System results>

7.3 Comparison with Other Systems

We compared our system with other three Chinese word segmentation systems:

1. The **CTG** system is a rule-based system with a very large dictionary containing XXX entries. It supports Chinese word segmentation and NER. CTG is developed at Microsoft Corporation. It is one of the best available products and delivers the best commercial accuracy today.
2. The **SR** system is a rule-based system with a base dictionary containing XXX entries. It supports Chinese word segmentation, factoid detection, NER, and partial morphological analysis. SR is developed by Beijing Language and Culture University. It is one of the best research systems in mainland China and achieved the top-rank performance in Chinese 863 evaluation.
3. The **NLPWIN** system is a Chinese parser, so linguistic knowledge such as syntactic structure can be explored for Chinese word segmentation, factoid detection, NER and morphological analysis. Although the base dictionary of NLPWIN is of medium size, it is able to detect new words. It is developed at Microsoft Research. NLPWIN is widely used for tasks such as grammar checking and machine translation.

As mentioned above, we compared the performance of LSP with that of the above three systems only in terms of NER precision/recall and # of OAS in order to achieve a fair comparison. However, we found that due to the difference of NE spec across systems, it is still very hard to compare their results automatically. For example, 北京市 *bei3-jing1-shi4* 'Beijing city' is tagged as a location name in LSP whereas in NLPWIN, it is tagged as two words: a location name 北京 *bei3-jing1* 'Beijing' and a word 市 *shi4* 'city'. Even worse, some location names tagged in one system are detected as organization names in another. Therefore we have to manually check the results of these systems. To do so, we randomly pick 900 sentences from the test set. According to the tags in the standard test set, the selected test set contains XXX words including 329 PN, 617 LN, and 435 ON. We did not differentiate LN and ON in evaluation. That is, for LN or ON detected by a certain system, we only check the word boundary and ignore its type. The results are shown in Table XX. We can see that LSP achieved the best overall performance of NER whereas NLPWIN shows the best ability of resolving OAS. This is not difficult to interpret. The use of the combination of context model and different class models (for different NE) enables LSP to capture richer statistical features for NER than other rule-based systems. The OAS result of NLPWIN demonstrates the effectiveness of exploring linguistic knowledge for resolving word segmentation ambiguity.

<Table XX. Comparison results>

7.4 Naturalness Evaluation

Sproat et al. (1996) presented a method of comparing the performance of Chinese word segmentation systems with the judgments of several human subjects. We call this the naturalness evaluation. Following Sproat et al.'s work, we used the same 900-sentence test set described in Section XX. We asked five native speakers to segment the test set. Since we could not bias the subjects towards a particular segmentation and did not presume linguistic sophistication on their part, the instructions were simple: subjects were to mark all places they might plausibly pause if they were reading the text aloud.

In addition to the four systems: CTG, SR, NLPWIN, and LSP, we also involved a greedy algorithm (i.e. forward maximum matching, FMM) and the standard segmentation, SS for reference.

Precision and recall were used to compare judgments. Clearly, for judge *J1* and *J2*, taking *J1* as standard and computing the precision and recall for *J2* yields the same results as taking *J2* as the standard, and computing for *J1*, respectively, the recall and precision. We therefore used the arithmetic mean of each interjudge precision-recall pair as a single measure of interjudge similarity. Table XX shows these similarity measures. The average agreement among the human judges is 0.86, and the average agreement between LSP and the humans is 0.85, or about 99% of the interhuman agreement. Interestingly, the average agreement between SS and the humans is about 100% of the interhuman agreement because SS is also generated by humans.

From the comparison results shown in Sections XX and XX, we have some confidence that the performance of our system is among the best in both market and research community.

8. DISCUSSION

Error analysis

9. CONCLUSION

Reference